

Scalability Issues in P2P Systems

Edmundo de Souza e Silva, Rosa M. Leao, Daniel S. Menasche
Federal University of Rio de Janeiro, Brazil

Don Towsley
University of Massachusetts at Amherst, USA

Abstract

One of the most fundamental problems in the realm of peer-to-peer systems consists of determining their service capacity. In this paper, we first propose a new Markov model to compute the throughput of peer-to-peer systems. Then, we present a simple approximate model for obtaining the system throughput for large peer populations. From these models, we obtain novel insights on the behavior of p2p swarming systems that motivate new mechanisms for publishers and peers to improve the overall performance. In particular, we show that system capacity can significantly increase if publishers adopt the most deprived peer selection and peers reduce their service rate when they have all the file blocks but one.

1 Introduction

Peer-to-peer (P2P) swarming systems have been tremendously successful in disseminating content in the Internet and major companies have adopted this architecture. For instance, *Blizzard entertainment* distributes large files via the *Blizzard downloader*, which is based on the *BitTorrent Opensource*. Ubuntu Linux is also available to download via BitTorrent. The *Amazon Simple Storage Service* (Amazon S3) is an infrastructure for data storage that supports the BitTorrent protocol for distributing files. Although the fraction of P2P traffic has decreased in relative terms due to NetFlix (which contributes to the fraction of WWW traffic) it continues to grow in absolute terms and has increased approximately 12% between 2009 and 2011 [22]. In addition, recent proposals suggest the use of P2P for distributing large files in content oriented networks [10] paving the way towards a Network Swarm Architecture.

P2P architectures have been studied for over a decade and numerous models have been proposed

to address the performance of the approach. Examples include models to determine the impact of incentive mechanisms and the download/upload rates on performance [4, 7]; studies to understand *free riding* [27] and the way by which files are disseminated; studies on fairness and availability issues [6, 18]. In addition, several works have focused on block distribution and peer selection strategies [13].

Despite numerous papers in the area, scalability issues have only recently been addressed. A system is said to be scalable if the system's throughput, *i.e.*, the rate at which users complete their downloads, increases linearly with increasing user population. P2P systems have been thought to be scalable since each new user joining the system brings additional resources to it. Consequently, the total capacity available is expected to increase in proportion to the newly incorporated resources and, accordingly, the system's throughput should increase linearly and unboundedly with population size. However, intrinsic limitations of the P2P architecture to scale in this manner have been observed. Recently, Hajek and Zhou [8] showed that, when peers in a swarm and publisher adopt the random peer/random useful block policy, the system becomes unstable, that is, swarm population grows without bound if the peer arrival rate λ exceeds the server capacity (U) dedicated to that swarm. Briefly, this instability problem occurs because, when two peers meet, they may not have useful data to share. Understanding the stability region of peer-to-peer swarming systems has recently gained attention [8, 15, 16, 33], and unstable scenarios have been observed in practice [18].

In a peer-to-peer swarming system, each peer makes two decisions before transmitting a block: (a) which block to transmit and (b) to whom to transmit it. Although the former decision has received some attention in previous works (for instance, it has been shown that rarest-first block selection and random useful block selection yield the same stability region [8]), the implications of the peer selection strategy on throughput have not been thoroughly studied (notable exceptions being [15, 16]). Previous works have assumed peers choose their neighbors using random peer selection [8, 20, 33].

In this work we propose models to evaluate the impact of different system parameters and system strategies on attainable throughput. First, we derive an upper bound on the throughput achieved when the publisher adopts most deprived peer selection and rarest-first block selection, while peers adopt random peer selection and random useful block selection. The bound is significantly larger than the maximum attainable throughput in the scenarios studied in [8], where both peers and

publishers adopt random peer and random useful block selection. This means that the system's performance can substantially increase when the server adopts a simple policy that gives priority to those peers that possess the smallest number of blocks in the swarm.

We identified two regions in which the P2P systems can operate. In the first, where $\lambda < \lambda_s$, the system is stable. In the second, where $\lambda_s < \lambda < \lambda_c$, the system is unstable, but its throughput can still increase as a function of the number of peers in a swarm. This property that throughput increases with swarm size permits the development of efficient models capable of solving systems with a large number of users.

From the models we also propose:

- a new very simple and incentive-compatible policy adopted by peers, wherein peers reduce their service capacity when they possess all blocks but one (see Section 3), that produces very large performance gains when the system is near saturation. That is, by employing this new policy, the system can accommodate many more users than otherwise;
- a new simple policy adopted by the tracker, wherein the tracker protect newcomers to be preferably served by the publisher (see Section 4), that can also result in significant performance gains.

The remainder of this paper is organized as follows. First, in Section 2 we set the background, discuss the scalability problem and compare several peer and server policies. In Section 3 we describe the models used to obtain the results we present. Results obtained from the models are discussed in Section 4. We show how the system throughput scales with swarm population size and study the performance gains of the policies we propose. Related work is presented in Section 5 and Section 6 concludes our work.

2 Scalability of Peer-to-Peer Systems

In P2P systems every new peer entering a swarm brings additional resources (transmission capacity and storage) since it acts both as a client and as a content provider. Therefore it is natural to expect the overall system throughput to increase as the number of peers increases. However, the throughput growth is not unbounded. Figure 1, obtained from one of the models we developed, illustrates this well. The figure shows that throughput grows almost linearly in the size of the

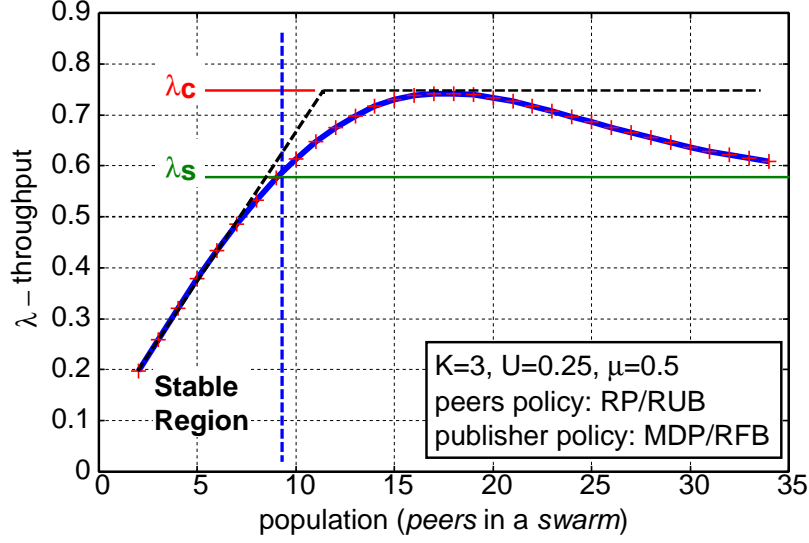


Figure 1: Limited throughput as peer population grows.

swarm population until it reaches a threshold (λ_c). Briefly, although the available resources increase with increasing number of peers, they are not fully utilized. For instance, this is the case where a peer has no block different from those that its neighboring peers possess.

We have observed two operating regimes in the P2P systems of interest to us. The first is a *stable* operating regime where all peers that arrive acquire the content in a finite amount of time. Associated with this regime, we have a threshold λ_s such that the system is stable provided that the peer arrival rate λ satisfies $\lambda < \lambda_s$. Then, our study points out that it is interesting to implement admission control, as in a closed system there is a second regime where the maximum achievable throughput, λ_c , can be larger than λ_s (see Figure 1).

We briefly explain the shape of the throughput curve of Figure 1. Assume a P2P system in which peers leave as soon as they complete downloads. The file downloaded by a particular swarm is divided into K equal size blocks. Let U be the server capacity *that is allocated to the swarm* in blocks per time unit. Let μ be the peers capacity. When the arrival rate of peers is large relative to U the system will reach, with high probability, a state in which all peers have all but one of the blocks (say, block X). In this scenario, only the server has block X and as soon as a peer obtains the missing block X from the server it leaves the system without helping to serve block X to others. On the other hand, new peers that arrive are quickly served by their peers and are able to obtain all blocks but X . As a consequence, the system behaves approximately like a client server system: peers depart at rate U since they do not cooperate to obtain the missing block. In addition,

although the available resources grow with the swarm size, these resources are not fully utilized.

Figure 2, also obtained from our analytical model, may be used to further clarify the behavior described above. We consider $K = 3$ blocks and a population of 15 peers. Assume that initially all peers have no blocks and that a new peer arrives as soon as another finishes downloading the desired file (closed system). Let T be the random variable equal to the time it takes until 90% of the peer population has all the blocks except the rarest. Figure 2(a) shows the cumulative distribution function of T . We see that $P(T \leq 50) > 0.9$ for the policies considered. For comparison purposes, note that by $t = 50$ approximately 25 (resp., 50) peers departed, on average, when $U = 0.5$ (resp., $U = 1.0$).

Once the system reaches the undesirable state where most peers have all but the rarest block, it takes considerable time to reach a state where the system does not behave as a client server system. This is shown in Figure 2(b). In this case, assume that at $t = 0$ all peers have all blocks but the rarest (except a single peer that has no blocks). Let L be the random variable equal to the time it takes until 50% of the peers leave the undesirable state, *i.e.*, until less than 8 peers have all blocks except the rarest. Figure 2(b) shows the cumulative distribution function of L . At $t = 50$, the probability that less than 8 peers have all blocks except the rarest is less than 10^{-4} for all policies considered.

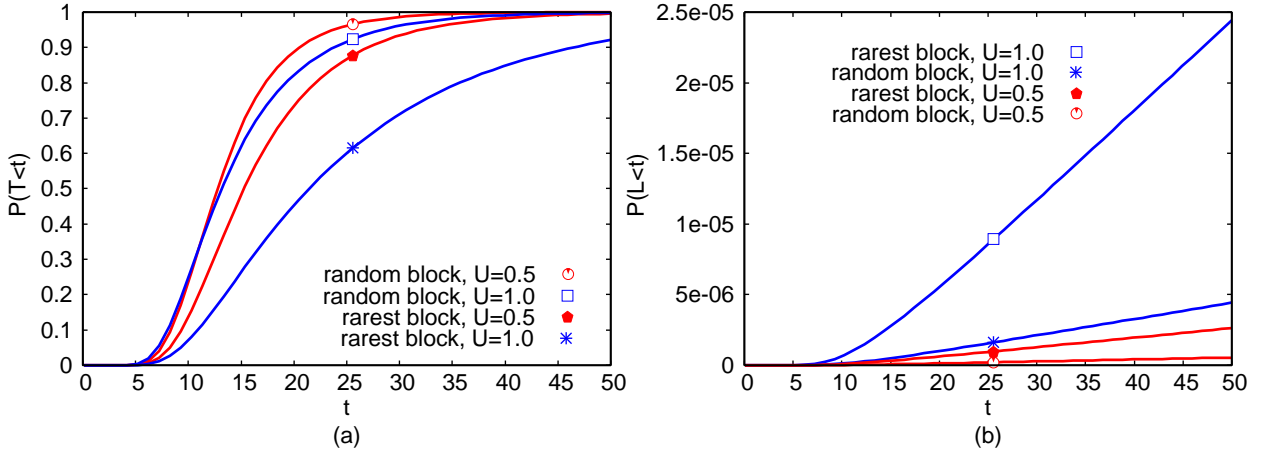


Figure 2: Peers evolving with time in a swarm: distribution of time until (a) all peers enter the one-club and (b) half population leaves the one-club. The publisher selects peers uniformly at random and its block selection policy varies. Peers adopt the random peer/random block policy ($\mu = 1.0$).

The queueing model we propose in Section 3 to estimate the limiting throughput takes into account that, when the system is saturated, with high probability most peers in the system eventually obtain all but the rarest block. We also show that the limiting throughput depends on the service

peers	publisher
random peer/random useful block (RP/RUB)	random peer/random useful block (RP/RUB)
random peer/rarest first block (RP/RFB)	random peer/rarest first block (RP/RFB)
random useful peer/random useful block (RUP/RUB)	most deprived peer/rarest first block (MDP/RFB)
random peer/random useful block (RP/RUB)	most deprived peer/rarest first block (MDP/RFB)

Table 1: Neighbor and block selection policies

policies adopted by the server and peers.

In a P2P system, the publisher needs to decide what block to transmit next and to which peer. Likewise, a peer has to decide what peer to contact and which block to request. In the literature, it has been assumed that peers go through random encounters and that peers are paired uniformly at random [8, 21, 30]. Nonetheless, if the publisher can strategically select its neighbors, then it is possible to improve the overall system capacity. Before introducing our models we first describe the policies that we consider.

Throughout this paper, we assume that peers select a neighbor uniformly at random at every transmission opportunity to exchange blocks (random peer selection). We also consider the case where trackers dynamically inform each peer P of the set of peers in the swarm in need of blocks owned by P . In this case, peers can select their neighbors uniformly at random among those that need the blocks they possess. We refer to this neighbor selection policy as random *useful* peer selection.

After choosing a neighbor, each peer selects one of its blocks for transmission to the neighbor. If the block is selected uniformly at random, the policy is referred to as random useful block selection. If peers have access to a list of the number of replicas of each block, they can build a rarest-block set containing the indices of the blocks with the least number of copies in the swarm [11]. This set can then be used by peers to select which block to transmit. This policy is referred to as rarest first block selection.

The publisher can select its peers and blocks in the same way as the peers. In addition, the publisher can also select its peers using the most deprived policy. Under this policy, the publisher prioritizes sending blocks to peers that own the least amount of blocks among those in the swarm. If the arrival rate of peers is large (or the swarm size is large) these peers are likely to be content-less peers, also referred to as newcomers. Table 1 summarizes these policies.

3 Models

In this section we present models of the P2P systems utilizing policies introduced previously as well as those we propose in this paper. Our results are based on these models. We first develop a Markov model that allows us to obtain throughput as a function of the number of peers in the swarm for different server and peer policies (Table 1). As the cardinality of the state space in the model becomes unmanageable for large swarms, we develop an approximate queueing model to calculate λ_s for large peer population sizes.

The policy where peers and publisher adopt random peer and random block selection has been studied in [8, 33]. These works show that the system is stable if and only if $\lambda < U$. In what follows, we show that a simple modification to the peer selection policy adopted by the publisher can produce a drastic increase in throughput. In addition, if a given set of peers reduce their upload rates, the maximum system throughput can further be significantly increased. Using our models we show how the limiting throughput depends on different system parameters.

3.1 Markov Model

We present a Markov model for calculating system throughput for different server and peers policies as a function of the swarm population. The model implements the fundamental characteristics of the policies studied here and is used to support the findings of this work. Due to the details included in the model, solving it numerically is limited to a relatively small number of blocks and moderate population sizes. Nevertheless, the results clearly support our main conclusions and provide the basis for an approximation that we propose in Section 3.2. Next, we present an overview of the Markov model. For a detailed description see Appendix A.

We model a fixed population system (closed system). As such, whenever a peer leaves the swarm a newcomer immediately arrives. Therefore, it approximates the behavior of swarms whose populations are approximately constant and allows us to study the system’s bottleneck as the population size grows. Although closed and open systems exhibit different behavior (e.g., [24]), the closed system provides insights on the stability region of the corresponding open system, because the closed system characterizes the open system at its saturation point where every departure triggers an arrival.

Let N be the number of peers in the swarm. A naive choice of state is to track the blocks each peer possesses. Clearly, this leads to a state space explosion and is intractable even for small peer population size and number of blocks. However, the model has symmetries that can be taken into account resulting in states that can be *lumped* (see Appendix C). Let Ω denote the model state space. One such symmetry allows us to choose the following state. Each state $\sigma \in \Omega$ is a vector $\sigma = (\sigma_1, \dots, \sigma_M)$ where σ_i is equal to the number of peers with signature i and $M = 2^K - 1$ is the total number of signatures. Other symmetries allow for additional reductions in the size of the state space of about one order of magnitude. Due to space limitations we briefly discuss such less evident symmetries, using a simple example. Assume the file contains 3 blocks. Each of these blocks may become the rarest. Suppose that R peers have all blocks except the first and $N - R$ peers have only the rarest block. It is not difficult to verify that this state can be lumped with similar states in which the rarest block is the second and the last. Our Markov model takes into account the states that can be lumped before their generation.

Figure 3 plots the throughput obtained from this model for different publisher and peer policies in Table 1. Although not all possible combinations of policies in Table 1 are shown in Fig. 3, the conclusions we obtain from the figure are valid in a broader set of settings. In the figure, the capacity of the publisher is one block per time unit, the peers download rate is 0.5 and the number of blocks is three. From the figure it is evident that the largest throughput values are obtained when the

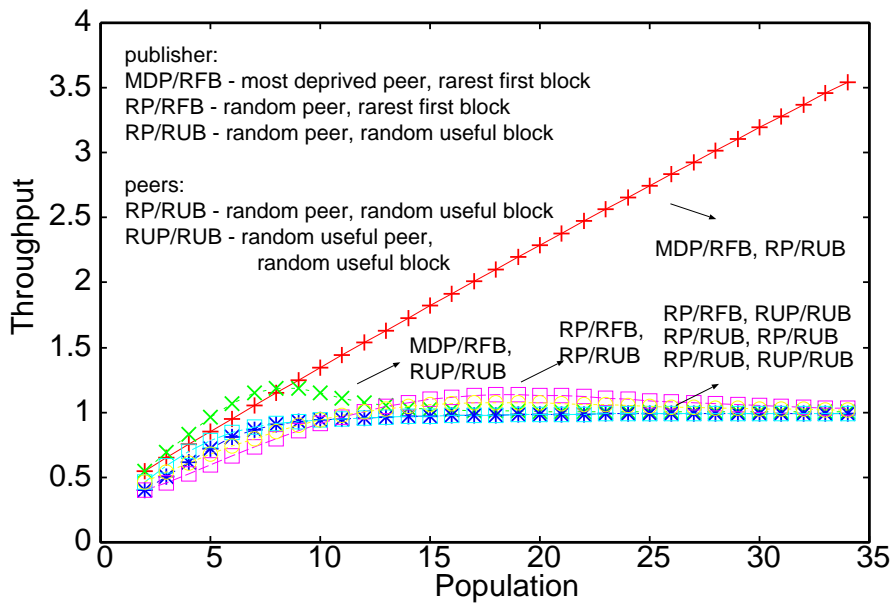


Figure 3: Comparing different policies.

publisher adopts the MDP/RFB and peers use the RP/RUB policies, for moderate to high swarm population sizes. This is true for a wide range of parameter values. Therefore, we choose to focus our results on this policy in Section 4. The advantages of the MDP/RFB policy adopted by the publisher will be quantified in what follows.

3.2 Approximate Queueing Model

Despite the computational savings generated by the existing symmetries of the Markov model to reduce the state space cardinality, solution of the model is impractical for large values of N and K . In this section we develop a queueing network model for calculating the limiting throughput for large population sizes. The model captures the main characteristics of the studied system policies and yet its simplicity also provides a better understanding of the fundamental effects of the parameter values on the system performance.

3.2.1 Overview of Peers and Publisher Dynamics

We start by selecting the publisher policy in which the server gives priority to peers with the smallest number of blocks and serves the rarest block to the chosen peer. That is, the server adopts the MDP and RFB policy (Table 1). In addition, peers adopt the RP/RUB policy, wherein upon a random peer contact, the block to be downloaded is chosen randomly from among those that the recipient peer does not have. We also assume that a peer can *reduce* its upload rate as follows: When a peer obtain $K - 1$ blocks it changes its service capacity from μ to μ' where $\mu' < \mu$. The motivation for this rate reduction will be clarified below when we show that it significantly increases the system throughput.

We refer to Figure 4 to describe the model. In this figure we identify five boxes each representing

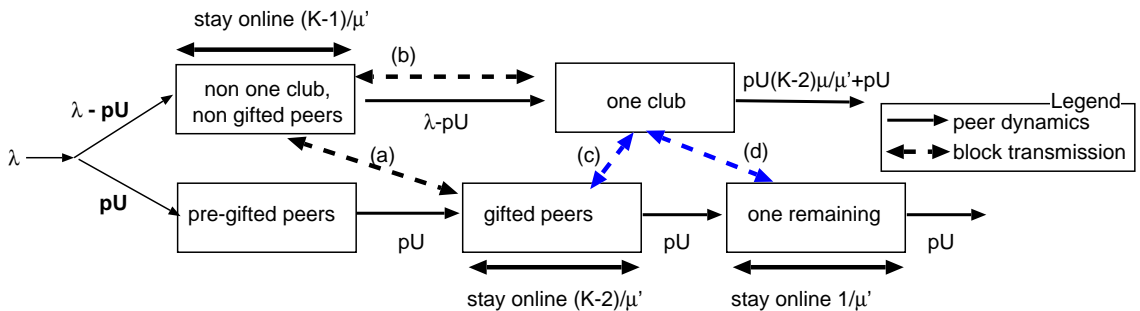


Figure 4: Flow dynamics

a set of peers with a given collection of blocks from the downloaded file. We assume that the peer arrival rate λ is sufficiently large, $\lambda > \lambda_c > U$, as in Figure 1. As argued in Section 2, the system eventually evolves to a state in which a large fraction of the peers have all but one of the blocks of the file being downloaded and this missing block is evidently the rarest. These peers are referred to as belonging to the *one-club* set [8] (see Figure 4).

Since the publisher adopts the most deprived peer selection and $U < \lambda$, a fraction pU/λ of peers receive a block from the publisher after arriving to the system. Note that: (a) newcomers are content-less, so they will be serviced with higher priority by the publisher and; (b) since λ is much larger than U , the publisher finds a newcomer with high probability as soon as it is ready to serve. Parameter p models the fact that the newcomers may not receive all of the publisher capacity available to the swarm. Due to the publisher RFB policy, newcomers receive the rarest block from the server. Peers that obtain the rarest block are called *gifted peers*. The first two boxes in Figure 4 represent peers in the process of obtaining the rarest block and those peers that already received it.

Remark: $p = 1$ corresponds to the server always serving the rarest block to a newly arrived peer if the server is available upon arrival. This occurs if newcomers are not immediately advertised to other peers, being known for some time only to the tracker (we will return to this issue in Section 4). If $p < 1$ the server and other peers *compete* for service and the newly arrived peer may obtain other blocks before obtaining the rarest block from the server. In this last case, newcomers obtain the rarest block from server with probability p , which depends on U and μ .

Peers adopt the random peer and random useful block selection policies, and transmission opportunities for each peer occur at their upload rate. Since the *gifted* set is relatively small compared to the *one-club* (system is saturated), the remaining peers that arrive will receive popular blocks (we call *popular* any block that it not the rarest) from the one-club set at rate μ' . This is true since peer encounters occur at random and the one-club set is large compared to the other sets of peers shown in Figure 4. For the same reason, the *gifted* peers transmit the rarest block to the *one-club* set and the remaining transmissions can be neglected (e.g., arrow (a) in Figure 4).

3.2.2 Queueing Network Model

The flow dynamics shown in Figure 4 motivate the development of the queueing network model presented next. Figure 5 illustrates the fixed population model that focuses solely on peers that do

not belong to the *one-club*. A fundamental model assumption is that the system is saturated and consequently, as argued previously, the number of peers belonging to the *one-club* is very large and approaches infinite as $N \rightarrow \infty$. We then track the behavior of peers that complete their download as they return for a new copy of the content. That is, we couple each completion with a new arrival (closed system model).

The model consists of $2(J+1)$ queues. Referring to Figure 5, the queues at the top are labeled

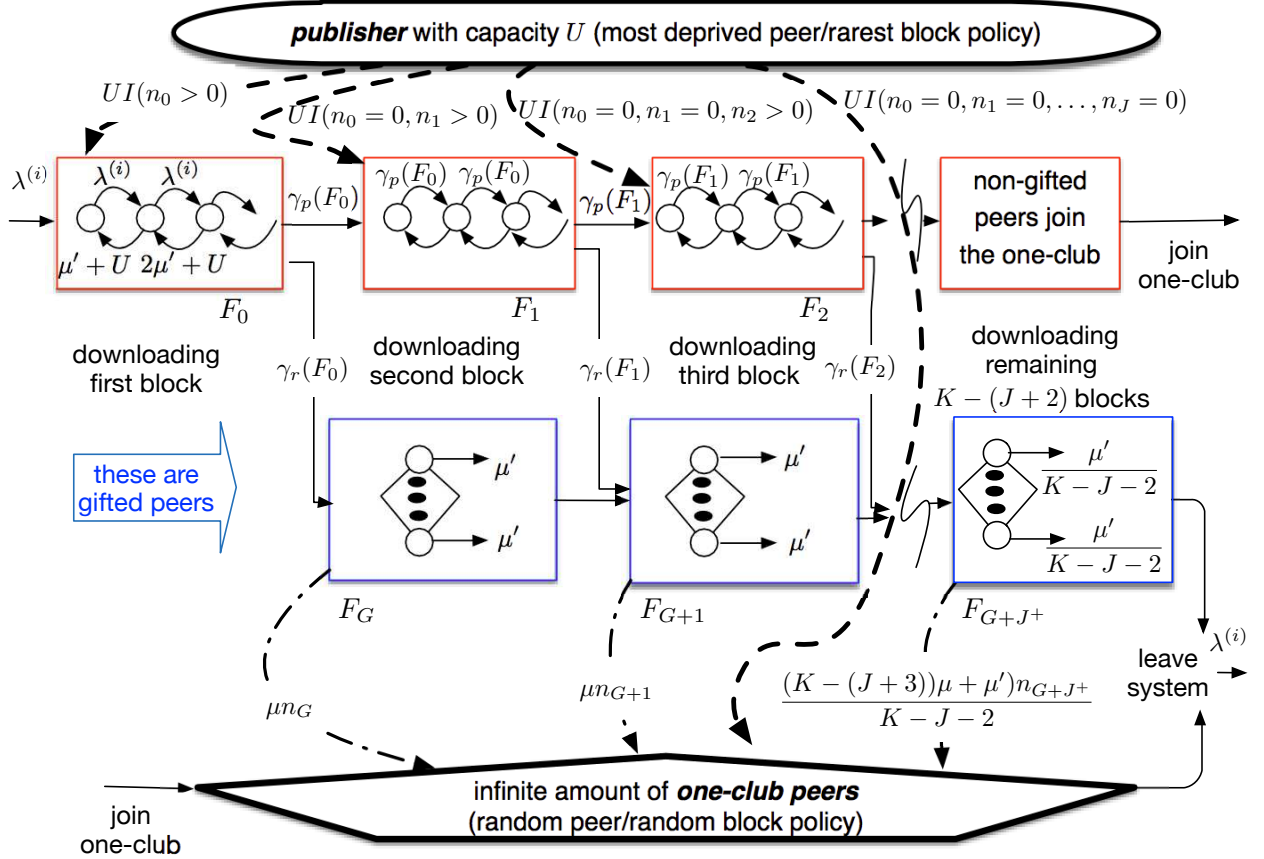


Figure 5: Queueing network model.

F_0, F_1, \dots, F_J and peers at these queues can be served both by the publisher and other peers. J is a model parameter to be chosen, $0 \leq J \leq K-2$. Model accuracy increases with J but, in turn, the computational complexity to solve the model also increases. In our experiments $J = 1$ was sufficient to achieve very good results. Each of these queues is modeled as a birth-death process with constant arrival rate and state dependent service rate as explained below. Queue F_0 represents newcomers and so peers in F_0 have no blocks. Queues F_1, \dots, F_J represent peers that have already obtained $1, \dots, J$ popular blocks, respectively, and are waiting to complete the download of an additional block. In order to further simplify the model, it is convenient to represent all peers with more than

J popular blocks collectively in a single group. This is the role of the rightmost box after queue F_2 in Figure 5. (These peers have no effect on the model, since they eventually join the *one-club*, which is assumed to be infinitely large.) Queue F_G represents all peers that downloaded only the rarest block and $F_{G+1}, \dots, F_{G+J-1}$ represent those peers that possess the rarest plus $1, 2, \dots, J-1$ popular blocks, respectively. In addition, the rightmost queue in the bottom of the figure (F_{G+J+}) represents all peers that have the rarest block plus J or more popular blocks.

Newcomers arrive with rate $\lambda^{(0)}$ and join queue F_0 . Since these peers have no blocks they can be served either by the publisher (that preferentially serves peers with the least number of blocks) or by other peers. Because the *one-club* is relatively large by assumption and encounters happen at random, the set of all *one-club* members serves a block to a tagged newcomer at rate μ' .

Let n_0 be the number of peers at queue F_0 . From the above arguments, the total service rate of F_0 is $n_0\mu' + U$ and the rate at which newcomers receive the rarest block is,

$$\gamma_r(F_0) = U(1 - \pi_0(F_0)), \quad (1)$$

where $\pi_0(F_0) = P(n_0 = 0)$ is the probability that queue F_0 is empty. Note that the rarest block can only be supplied by the publisher, due to the assumption that the *one-club* is infinite in size and the random peer selection policy adopted by the peers. Likewise, the rate at which newcomers obtain a popular block is

$$\gamma_p(F_0) = E[n_0]\mu', \quad (2)$$

where $E[n_0]$ is the expected queue length of F_0 .

If a newcomer gets the rarest block from the publisher, it proceeds to queue F_G . On the other hand, if the newcomer gets a block from the *one-club*, it joins queue F_1 . Since peers in F_1 have only one popular block, they are eligible to be served by the publisher, provided that queue F_0 is empty (most deprived policy). Similar to those peers in F_0 , those in F_1 can also be served by the *one-club*.

Gifted peers in F_G can only be served by *one-club* members, since the publisher gives preference to those peers that do not have the rarest block. Queue F_G is then modeled as an $M/M/\infty$ queue with aggregate departure rate $\mu'n_G$ (where n_G is the number of peers in F_G). Once peers in F_G obtain a popular block, they move to queue F_{G+1} to get an additional (popular) block. Finally,

after downloading the rarest block and $(J - 1)$ popular blocks peers move to queue F_{G+J+} , which models the peers with more than $(J + 1)$ blocks, one of them being the rarest.

If a peer in F_1 gets a block from the publisher, it moves to queue F_{G+1} , which models peers that have one popular block in addition to the rarest block. Otherwise, a peer in F_1 moves to F_2 . Note that we assume that the probability that peers with more than J popular blocks obtain the rarest block from the publisher is very small. This is true since the publisher gives priority to peers with the least number of blocks. Therefore, if a peer obtains one additional popular block after reaching F_J it eventually joins the *one-club* when it finishes downloading the remaining $K - (J + 2)$ blocks.

We should note that the service rate of queue F_j ($j = 1, \dots, J$) depends on the states of queues F_0, \dots, F_{j-1} . That is, the service rate of F_j is $\mu' + UI[n_0 = 0, \dots, n_{j-1} = 0]$, where $I[X]$ is an indicator function equal to 1 when the event X occurs (in this case X is the event $\{n_0 = 0, \dots, n_{j-1} = 0\}$). As an approximation, we break this dependency by using the independent stationary value $P(n_j = 0)$ of each queue F_i , $i < j$. Therefore, the rate at which peers in F_j are served is approximated as $\mu' + U \prod_{i=0}^{j-1} \pi_0(F_i)$. From this, we obtain the rate at which peers in queue F_j move to F_{G+j} :

$$\gamma_r(F_j) = U \left(\prod_{i=0}^{j-1} \pi_0(F_i) \right) (1 - \pi_0(F_j)), \quad (3)$$

where the product in parenthesis is defined as 1 when $j = 0$. The rate at which peers in F_j move to F_{j+1} is:

$$\gamma_p(F_j) = E[n_j] \mu'. \quad (4)$$

Once the arrival rate at each queue is obtained we compute the rate at which the *one-club* is served. We observe that peers in queues $F_G, \dots, F_{G+J}, F_{G+J+}$ possess the rarest block and, therefore, they can serve peers in the *one-club* at rate μ or at rate μ' when peers obtain $K - 1$ blocks. Since F_{G+J+} models peers that have more than J blocks and because this is an $M/M/\infty$ queue, the expected number of peers that have the *rarest* block plus $J, J + 1, \dots, K - 2$ blocks are all identical and a fraction $1/((K - 1) - J)$ of n_{G+J+} have $(K - 1)$ blocks and serve the *one-club* at rate μ' . (Refer to the dashed lines in Figure 5.) We can then calculate the rate Ψ at which the

gifted peers serve the rarest block to the *one-club* members,

$$\Psi = \frac{\mu}{\mu'} \sum_{l=0}^J \left((K-2-l) + \frac{\mu'}{\mu} \right) \gamma_r(F_l), \quad (5)$$

where $\gamma_r(F_i)$ is given by (3) (see Appendix B for the derivation of (5)).

The total rate at which peers leave the system, $\Gamma^{(0)}$, is calculated by summing the departure rate of the *gifted* peers and the rate at which the *one-club* is served by the *gifted* peers and by the publisher. We have:

$$\Gamma^{(0)} = \sum_{i=0}^J \gamma_r(F_i) + \Psi + U \prod_{i=0}^J \pi_0(F_i). \quad (6)$$

In order to solve the overall model, assume that the arrival rate $\lambda^{(0)}$ is known. The steady state solution for queue F_0 is obtained by constructing a simple birth-death process with parameter values dependent on U and μ' . After computing $E[n_0]$, we rely on the independence assumption mentioned above to obtain the arrival rate to queues F_G and F_1 . Similarly, the arrival rates to the remaining queues are calculated as well as the system departure rate $\Gamma^{(0)}$, using equation (6). The final solution is obtained by iterating $\lambda^{(n)} = \Gamma^{(n-1)}$ until convergence is achieved. In our experiments we found that $J = 1$ is sufficient to obtain accurate results and this was the value we used in the curves plotted in Section 4.

From the model we can calculate the limiting throughput when the publisher is able to use all its bandwidth on newcomers and is always busy.

Proposition 3.1 *When the server adopts the MDP/RFB policy, peers adopt the RP/RUB policy and newcomers are served by the publisher if it is available, the system throughput is limited by*

$$\lambda \leq (((K-2)\mu/\mu') + 2) U \quad (7)$$

We use the queueing network model to obtain (7). Because newcomers are served by the publisher whenever available, the server capacity U is totally devoted to the newly arriving peers. Therefore, $\Gamma_r(F_0) = U$, and $\Gamma_r(F_i) = 0$ for $i = 0, \dots, K-1$. (Here we assume $J = K-2$, but we could choose any value for J .) This means that peers in $F_1, \dots, K-2$ will join the *one-club*. In addition, the arrival rate to queues $F_G, F_{G+1}, \dots, F_{G+K-2}$ is U , since the server only serves the newcomers.

Then, equation (5) reduces to:

$$\Psi = \frac{\mu}{\mu'}(K-2)U + U \quad (8)$$

Substituting (8) into (6) yields

$$\Gamma^{(0)} = \frac{\mu}{\mu'}(K-2)U + 2U = \left(\frac{\mu}{\mu'}(K-2) + 2\right)U \quad (9)$$

If $\Gamma^{(0)} > U$ or, equivalently, $\mu/\mu'(K-2) + 2 > 1$, then $\lambda^{(1)} > U$ and the result follows as long as the server is never idle, or equivalently $\pi_0(F_0) = 0$. \square

This simple result can also be directly obtained by observing the flow dynamics shown in Figure 4. Proposition 3.1 assumes $p = 1$. The *gifted* peers spend, on average, $(K-2)/\mu'$ time units to obtain $(K-2)$ blocks from members of the *one-club* set. From Little's result, on average there are $U(K-2)/\mu'$ in the *gifted* peers set, i.e., *gifted* with no more than $(K-2)$ blocks and each peer in this set is serving the *one-club* at rate μ . After receiving the $(K-1)$ -th block, these *gifted* peers reduce their upload rate to μ' due to the adopted peer service policy. Therefore, the aggregate rate at which the *one-club* is served with the missing block is $\mu/\mu'(K-2)U + U$ (arrows (c) and (d) in Figure 4). Thus, the *one-club* departure rate is $\mu/\mu'(K-2)U + U$ and the overall system throughput is $((K-2)\mu/\mu' + 2)U$.

Comments:

- Zhou and Hajek [8] showed that the limiting throughput is U for the random peer random useful block selection policy. Proposition 3.1 indicates that, if the publisher adopts the most deprived peer and rarest first block policy, then the largest peer arrival rate that the system can support is $((K-2)\mu/\mu' + 2)U$. That is, with a minor change in the publisher policy, the achievable throughput can be significantly increased.
- The larger the number of blocks the larger is the maximum throughput. This is a powerful result and quantifies the advantage of dividing a file into small blocks. Clearly, one cannot reduce the block size towards zero, due to the inherent overhead introduced by headers associated to each block. Future works consists of using Proposition 3.1 to cope with the trade-off between decreasing the block size and increasing the overhead of headers and other control data that is transmitted per block.

- The proposition states a counter-intuitive result. If peers *reduce* their upload rate once they obtain all but one of the blocks, the system throughput increases inversely to μ' . Reducing μ' saves peer bandwidth, increases throughput and does not require any incentive mechanism! Note that equation (7) does not impose any limit to μ' . If μ' decreases towards zero, at some point the throughput may decrease. In Section 4, we indicate through a few examples that μ' can be made more than one order of magnitude smaller than μ , which is sufficient for large performance gains.

4 Results

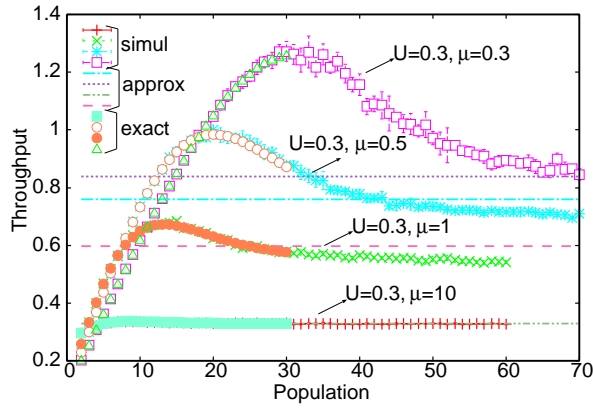
In this section we present numerical results obtained with the Markov model and the approximate queueing network model. Our goals are: (a) to show how the throughput varies with different model parameters; (b) to motivate the proposed server and peer policies.¹

4.1 Validating the Approximation

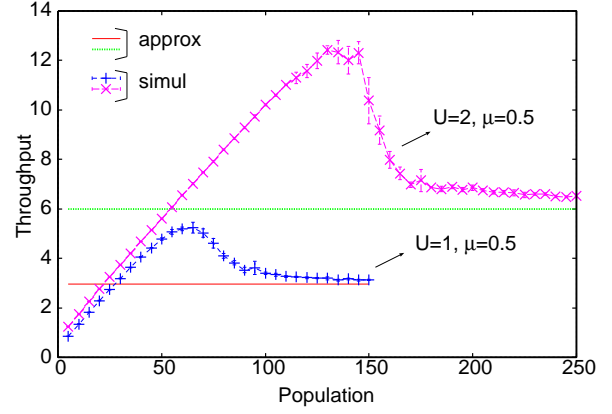
We consider two scenarios: $U \leq \mu$ and $U > \mu$, to validate the approximation for the throughput λ_s obtained from the queueing model. For both scenarios the file consists of three blocks and $\mu = \mu'$.

Figure 6(a) shows the case where $U \leq \mu$ and Figure 6(b) where $U > \mu$. In these figures three results are presented: throughputs obtained from the analytical solution of the Markov model (for a population from 2 to 30 users), throughputs obtained from simulation of the Markov model (for large populations) and throughputs computed using the queueing network model. Not surprisingly, the solutions obtained analytically are within the confidence interval of the simulation. For all values of U and μ , as the population increases, the throughput increases until it reaches a maximum value and then it decreases and tends to λ_s . From the figures we observe that the queueing model presented in Section 3 provides a good estimate of λ_s . The maximum relative error between the Markov model and the queueing model is equal to 10%.

¹Appendices D and E contain additional results on the transient system analysis and about peers that reside in the system after completing their downloads.



(a) $U \leq \mu$

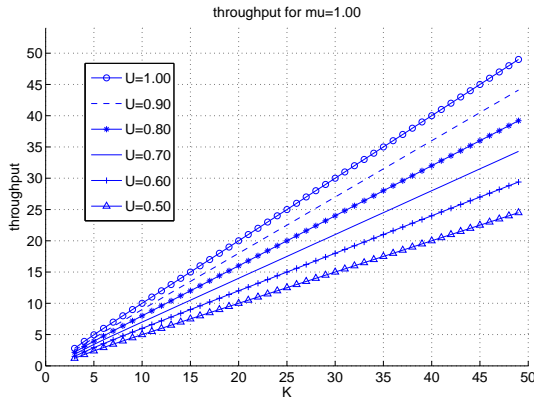


(b) $U > \mu$

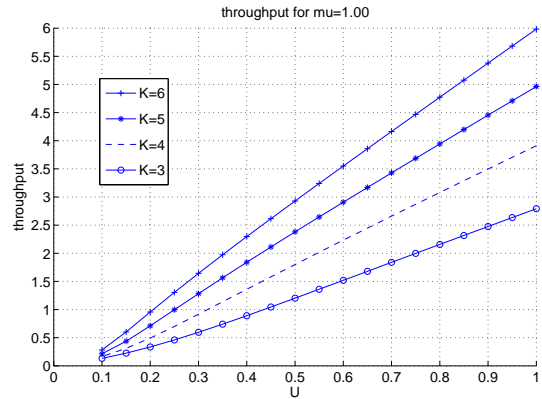
Figure 6: Validating the approximation, $K = 3, \mu = \mu'$.

4.2 Varying Service Capacity of the Server

Figure 7 illustrates how throughput varies with K and U . These results were obtained from the queueing model of Section 3. The figure shows that the model captures the fact that the throughput is approximately linear in K and U , for constant μ . This occurs because the majority of the server transmissions are for peers that do not have any block. Each gifted peer serves, on average, $(K - 1)$ one-club peers before leaving the system. Thus, for each block served by the publisher, roughly K peers leave the system ($(K - 1)$ from one-club and one gifted). As the server rate is U blocks per time unit, the system throughput grows linearly with K and U .



(a) Throughput as a function of K



(b) Throughput as a function of U

Figure 7: The throughput varies linearly with U and K .

4.3 Modifying Service of Peers with $(K - 1)$ Blocks

In this section we study system throughput when peers that have collected all blocks but one (one-club peers), serve with rate μ' , where $\mu' < \mu$. The motivation to decrease the upload rate of the one-club peers is to keep the gifted peers in the system longer serving the rarest block. This policy is motivated by the results presented in Section 3.2.1. They indicate that the throughput can increase if the upload rate of the one-club peers decreases.

Figures 8 and 9 show the increase of throughput when the one-club peers reduce their upload rate from μ to μ' . The throughput increases as μ' decreases. It is important to note that the throughput increases significantly with this simple modification in the service policy of one-club peers. The policy is very easy to implement and is incentive-compatible as it allows peers to save bandwidth.

However, the increase of throughput as the upload rate of the one-club peers decreases is not unbounded as $\mu' \rightarrow 0$. For small values of μ' (not shown in Figure 9(a)) the throughput decreases as we further reduce μ' . In particular, if $K = 2$ and $\mu' = 0$, the throughput is equal to $U/2$, since all blocks will be served only by the server (the system degenerates in a client server system).

Figure 9(b) shows the value of the throughput as the population increases. We note that the throughput is significantly higher when $\mu' < \mu$ for a population greater than 10 users, and as the population grows the gains in throughput increase.

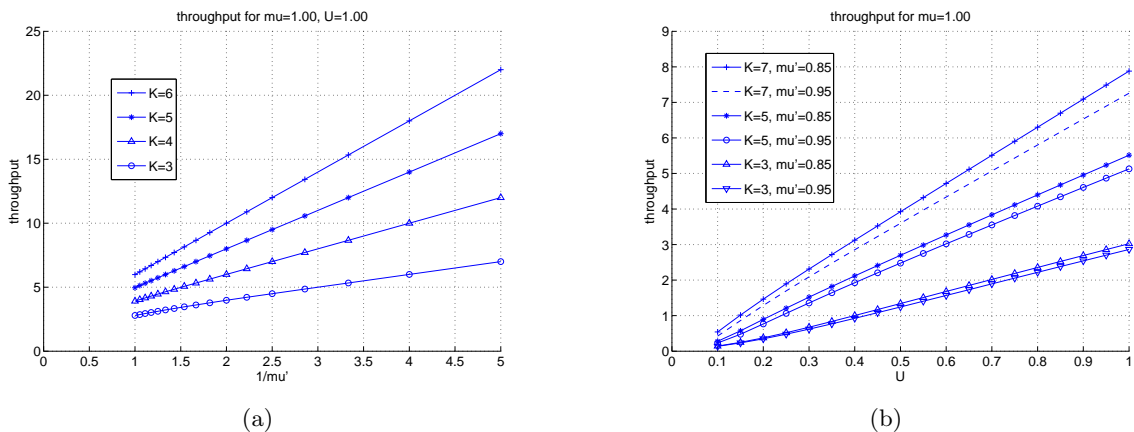


Figure 8: Throughput when the policy $\mu' < \mu$ is adopted by the one-club peers (K varying between 3 and 7): (a) as $1/\mu'$ and (b) U increase, the throughput increases roughly linearly.

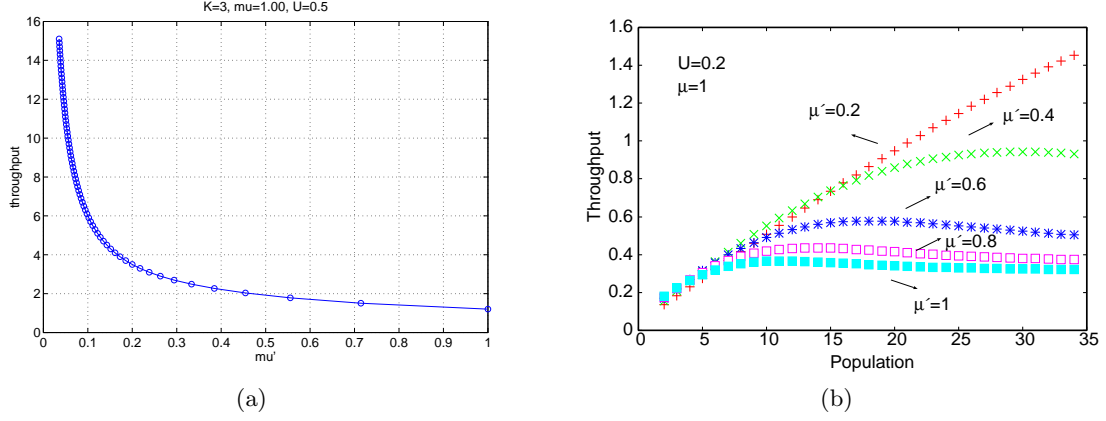


Figure 9: Throughput when the policy $\mu' < \mu$ is adopted by the one-club peers ($K = 3$).

4.4 Shielding Newcomers

In this section we show that *hiding* newcomers from other peers can significantly improve system throughput in some scenarios. Suppose the server rate U is less than the peer upload rate μ . In this case, newcomers receive the most popular blocks before the rarest block with high probability, and then, leave the system immediately after receiving the rarest block. The idea of *shielding newcomers* is to prevent newcomers from receiving the most popular blocks before receiving the rarest one. This policy is easily implemented by the tracker: it just does not announce the newcomers to other peers.

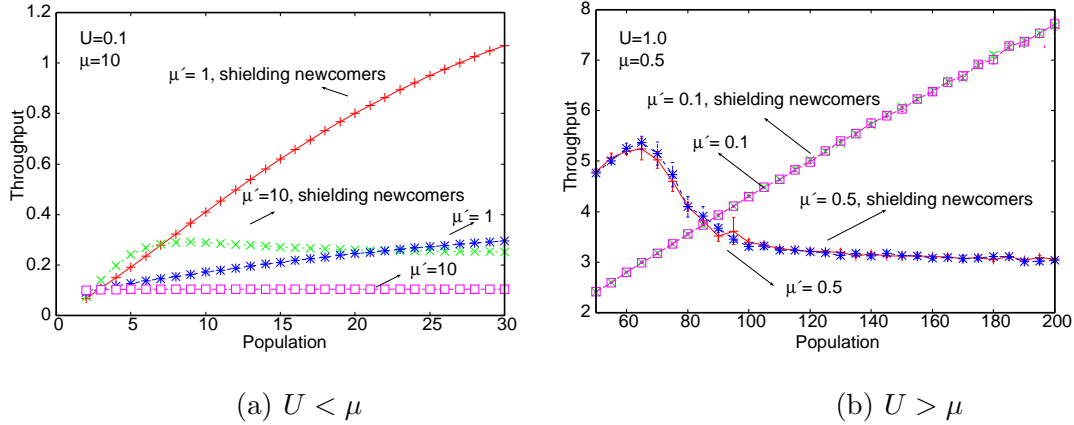


Figure 10: Throughput when combining the policies of shielding newcomers and setting $\mu' < \mu$.

Figure 10 shows the system throughput when $U < \mu$ and $U > \mu$, for $K = 3$. In Figure 10(a), we consider four scenarios: (i) $\mu = \mu' = 10$, (ii) $\mu = 10$ and $\mu' = 1$, (iii) $\mu = \mu' = 10$, shielding newcomers and (iv) $\mu = 10$ and $\mu' = 1$, shielding newcomers. In scenario (i), the one-club peers serve newcomers before they receive the rarest block from the server. In this case throughput converges to $U = 0.1$. In the second scenario, one-club peers reduce their upload rate and throughput increases

when compared to scenario (i), as expected. (see Section 4.3). The third scenario implements the policy of shielding newcomers. The throughput λ_s is approximately equal to 0.25. For a small population size, the throughput reaches the KU value established in Proposition 3.1. Scenario (iv) combines both policies: the upload rate of one-club peers is reduced and newcomers are shielded. Throughput significantly increases when these policies are used in combination. When the policies are used separately, the throughput is approximately 0.3 for a population of 30 users. When both policies are used, the throughput exceeds 1.0.

Figure 10(b) presents the throughput when $U > \mu$. The figure shows that the policy shielding newcomers is not necessary in this case. The throughput is roughly the same irrespective of whether the policy is used or not. When server capacity U is greater than the upload rate of the one-club peers μ , the server transmits the rarest block to the newcomers at a higher rate than the rate they receive the most popular blocks. Then, it is not necessary to *hide* the newcomers from the one-club peers because newcomers will spontaneously receive the rarest block from the publisher before receiving other blocks from the remaining peers.

5 Related Work

There is a vast literature on the stability and throughput of peer-to-peer swarming systems focusing on its relations with multiple swarms and bundling [6,32,34], self-sustainability [5], real-time content dissemination [2,31], coding [26] and system design [1,9,15,21,32]. Nonetheless, we were unable to find any previous work that accounted for the *missing piece syndrome* when computing the *throughput* of peer-to-peer swarming systems.

In previous works, authors assumed that either peers and publishers adopted random-peer selection [20], files had at most two blocks [19] or swarming systems behaved differently as compared to the system analyzed in this paper [12]. The most-deprived peer selection policy was first proposed by Bonald *et al.* [3]. As indicated in this paper, if the publisher adopts the most-deprived peer selection strategy, the throughput of the swarm can increase even if the remaining peers do not change their strategies.

Yang and de Veciana were the first to consider a closed system to analyze the transient increase in throughput after a flash crowd [28]. They also considered an idealized fluid model to study the

steady state. In their seminal paper, Yang and de Veciana did not account for the fact that a block might become rare and its retrieval turn into a system bottleneck.

In the peer-to-peer literature, fluid models have been traditionally used to study system performance [14,23,25] and scheduling strategies [29]. The importance of taking into account the fact that the file is divided into finite blocks rather than considering the fluid limit was indicated in [15,32]. In this paper, we compute the throughput of swarming systems accounting for the fact that the file is divided into finite blocks, and use our model to motivate novel scheduling strategies.

Scheduling strategies to improve system throughput usually rely on some sort of altruism. To improve system throughput, in previous works it has been proposed that peers reside in the system after completing their downloads [32], barter for content that they were not interested in [32] or refrain from taking advantage of all contact opportunities [21]. In this paper, in contrast, we show that a simple and incentive-compatible strategy, which consists of reducing the service capacity of the peers that have all but one block, can significantly improve system throughput.

6 Conclusions

Due to their ability to scale, robustness and efficiency, P2P systems are responsible for a significant portion of today's Internet traffic and constitute the basis for new architectures such as content centric networking [10]. Although P2P systems are very popular, their fundamental limitations are yet to be fully understood. In this paper we present new results to quantify the throughput of P2P systems which we hope will shed some light on the subject. Inspired by the presented models, we also propose new server and peer policies that result in significant performance improvements.

References

- [1] E. Altman, P. Nain, A. Schwartz, and Y. Xu. Predicting the impact of measures against p2p networks: transient behavior and phase transition. *TON*, 21(3):935–949, 2013.
- [2] F. Baccelli, F. Mathieu, I. Norros, and R. Varloot. Can p2p networks be super-scalable? In *IEEE INFOCOM*, 2013.
- [3] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg. Epidemic live streaming: optimal performance trade-offs. In *SIGMETRICS*, volume 36, pages 325–336. ACM, 2008.
- [4] A. Chow, L. Golubchik, and V. Misra. Bittorrent: An extensible heterogeneous model. In *INFOCOM*, pages 585–593. IEEE, 2009.
- [5] D. Ciullo, V. Martina, M. Garetto, E. Leonardi, and G. Torrisi. Stochastic analysis of self-sustainability in peer-assisted vod systems. In *IEEE INFOCOM*, pages 1539–1547, 2012.

- [6] E. de Souza e Silva, R.M.M. Leão, D.S. Menasché, and A.A. Rocha. On the interplay between content popularity and performance in P2P systems. In *QEST*, pages 3–21. Springer, 2013.
- [7] B. Fan, D. Chiu, and J. Lui. The delicate tradeoffs in bittorrent-like file sharing protocol design. In *ICNP*, pages 239–248. IEEE, 2006.
- [8] B. Hajek and J. Zhu. The missing piece syndrome in peer-to-peer communication. In *IEEE ISIT*, 2010.
- [9] K. Hwang, V. Gopalakrishnan, R. Jana, S. Lee, V. Misra, K. Ramakrishnan, and D. Rubenstein. Joint-family: Enabling adaptive bitrate streaming in p2p video-on-demand. In *ICNP*, 2013.
- [10] Van Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking named content. In *CONEXT*, pages 1–12. ACM, 2009.
- [11] A. Legout, N. Liogkas, E. Kohler, and L. Zhang. Clustering and sharing incentives in Bittorrent systems. In *ACM SIGMETRICS*, 2007.
- [12] L. Leskelä, P. Robert, and F. Simatos. Interacting branching processes and linear file-sharing networks. *Advances in Applied Probability*, 42(3):834–854, 2010.
- [13] Laurent Massoulié and Andrew Twigg. Rate-optimal schemes for peer-to-peer live streaming. *Performance Evaluation*, 65(11-12):804–822, 2008.
- [14] Laurent Massoulié and Milan Vojnovic. Coupon replication systems. *IEEE/ACM Transactions on Networking (TON)*, 16(3):603–616, 2008.
- [15] Fabien Mathieu and Julien Reynier. Missing piece issue and upload strategies in flashcrowds and p2p-assisted filesharing. In *AICT/ICIW*, 2006.
- [16] D. S. Menasché, A. A. de A. Rocha, E. de Souza e Silva, D. Towsley, and R. M. M. Leão. Implications of peer selection strategies by publishers on the performance of p2p swarming systems. *ACM SIGMETRICS Performance Evaluation Review*, 39(3):55–57, 2011.
- [17] Daniel S Menasché, AAA Rocha, E de Souza e Silva, Rosa MM Leao, and Don Towsley. Stability of peer-to-peer swarming systems. In *SBRC*, 2012. http://ce-resd.facom.ufms.br/sbrc/2012/ST4_1.pdf.
- [18] F. Murai, A.A. Rocha, D. Figueiredo, and E. de Souza e Silva. Heterogeneous download times in a homogeneous bittorrent swarm. *Computer Networks*, 56:1983–2000, 2012.
- [19] Ilkka Norros, Hannu Reittu, and Timo Eirola. On the stability of two-chunk file-sharing systems. *Queueing Systems*, 67(3):183–206, 2011.
- [20] R. Núñez-Queija and B. Prabhu. Scaling laws for file dissemination in p2p networks with random contacts. In *Quality of Service. International Workshop on*, pages 75–79. IEEE, 2008.
- [21] B. Oguz, V. Anantharam, and I. Norros. Stable, distributed p2p protocols based on random peer sampling. In *Allerton Conf. on Comm., Control and Comput.*, pages 915–919. IEEE, 2012.
- [22] J. S. Otto, M. A. Sanchez, D. R. Choffnes, F. E. Bustamante, and G. Siganos. On blind mice and the elephant – understanding the network impact of a large distributed system. In *SIGCOMM*, 2011.
- [23] D. Qiu and R. Srikant. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *SIGCOMM*, volume 34, pages 367–378. ACM, 2004.

- [24] B. Schroeder, A. Wierman, and M. Harchol-Balter. Open versus closed: A cautionary tale. In *NSDI*, 2006.
- [25] F. Simatos, P. Robert, and F. Guillemin. Analysis of a queueing system for modeling a file sharing principle. In *ACM SIGMETRICS*, 2008.
- [26] Cedric Westphal. A stable fountain code mechanism for peer-to-peer content distribution. *arXiv preprint arXiv:1401.5099*, 2014.
- [27] R.L. Xia and J. Muppala. A survey of bittorrent performance. *IEEE Communications Surveys & Tutorials*, 12(2):140–158, 2010.
- [28] X. Yang and G. de Veciana. Performance of peer-to-peer networks: Service capacity and role of resource sharing policies. *Performance Evaluation*, 63:175–194, 2006.
- [29] Bo Zhang, Sem C Borst, and Martin I Reiman. Optimal server scheduling in hybrid p2p networks. *Performance Evaluation*, 67(11):1259–1272, 2010.
- [30] X. Zhou, S. Ioannidis, and L. Massoulié. On the stability and optimality of universal swarms. *ACM SIGMETRICS Performance Evaluation Review*, 39(1):301–312, 2011.
- [31] J. Zhu and B. Hajek. Tree dynamics for peer-to-peer streaming. *arXiv preprint arXiv:1308.1971*, 2013.
- [32] Ji Zhu. *Stability and performance in peer to peer networks*. PhD thesis, University of Illinois at Urbana-Champaign, 2014.
- [33] Ji Zhu and Bruce Hajek. Stability of a peer-to-peer communication system. *Information Theory, IEEE Transactions on*, 58(7):4693–4713, 2012.
- [34] Ji Zhu, Stratis Ioannidis, Nidhi Hegde, and Laurent Massoulié. Stable and scalable universal swarms. In *PODC*, pages 260–269. ACM, 2013.

Appendix A: Markov Model Details

We model a swarm as a continuous-time Markov chain with state space Ω and infinitesimal generator Q . Let $\mathcal{F} = \{1, \dots, K\}$ and \mathcal{C} be the set of subsets of \mathcal{F} .

Each user has a signature, defined as a set containing element i if the user has block i and 0 otherwise, $i = 1, 2, \dots, K$. As users leave the system as soon as they obtain their last block, each user has one of $2^n - 1$ signatures.

Let σ_C be the number of peers with signature C , where $C \in \mathcal{C} \setminus \mathcal{F}$. State $\sigma \in \Omega$ is characterized by the number of users with each signature, $\sigma = (\sigma_\emptyset, \sigma_{\{1\}}, \dots, \sigma_{\mathcal{F} \setminus \{K\}})$. Note that it is possible to lump the state space, but to simplify presentation in this appendix we consider the unlumped state space. Let \mathbf{e}_C denote the vector with the same dimension as σ , with a one in position C and other coordinates equal to zero.

Let $\Gamma_s(C, C')$ and $\Gamma_p(C, C')$ be the aggregate transition rate of peers of type C to type C' due to service from the server and from other peers, respectively. We assume peers adopt the random peer, random useful block selection, whereas the publisher strategy is varied. In some cases, it will be convenient to make explicit the block which is received by a peer with signature C , replacing C' by C'_j when block j is received. After a peer with signature C receives block j , $j \notin C$, the number of peers with signature C'_j increases by one. Recall that as soon as peer completes its download a new one arrives (closed system). Then,

$$C'_j = \begin{cases} C \cup \{j\}, & \text{if } |C| < K - 1 \\ \emptyset, & \text{otherwise} \end{cases} \quad (10)$$

Next, we characterize the positive elements of Q . When a peer that has all blocks except j gets block j , its signature transitions from $\mathcal{F} \setminus \{j\}$ to \mathcal{F} . Then, it immediately leaves the system and another peer, with signature \emptyset , arrives. Then, the rate at which the system transitions from state σ to $\sigma - \mathbf{e}_{\mathcal{F} \setminus \{j\}} + \mathbf{e}_\emptyset$ is,

$$q_{\sigma, \sigma - \mathbf{e}_{\mathcal{F} \setminus \{j\}} + \mathbf{e}_\emptyset} = \Gamma_s(\mathcal{F} \setminus \{j\}, \emptyset) + \Gamma_p(\mathcal{F} \setminus \{j\}, \emptyset), \text{ for } j = 1, 2, \dots, K \quad (11)$$

When a peer that needs more than one blocks gets block j , its signature transitions from C to $C \cup \{j\}$.

The corresponding rate at which the system transitions from state σ to state $\sigma - \mathbf{e}_C + \mathbf{e}_{C \cup \{j\}}$ is

$$q_{\sigma, \sigma - \mathbf{e}_C + \mathbf{e}_{C \cup \{j\}}} = \Gamma_s(C, C \cup \{j\}) + \Gamma_p(C, C \cup \{j\}), \text{ for all } C \in \mathcal{C}, |C| < K - 1 \quad (12)$$

Let π_σ be the steady state probability of state σ . The vector of steady state probabilities is denoted by π .

Let λ be the throughput of the Markov model. The throughput is given as a function of $q_{\sigma, \sigma - \mathbf{e}_{\mathcal{F} \setminus \{j\}} + \mathbf{e}_\emptyset}$ as follows,

$$\lambda = \sum_{\sigma \in \Omega} \sum_{j=1}^K \pi_\sigma q_{\sigma, \sigma - \mathbf{e}_{\mathcal{F} \setminus \{j\}} + \mathbf{e}_\emptyset} \quad (13)$$

Let \mathcal{N}'_S be the set of neighbors of each of the peers with signature S , *i.e.*, the candidate peers to which a peer with signature S can potentially transfer content. Except otherwise noted, we assume $|\mathcal{N}'_S| = N - 1$.

Let μ_S be the service capacity of peers with signature S . When considering homogeneous peers, we let $\mu_S = \mu$ for all $S \in \mathcal{C}$. When studying the special policy wherein peers with all blocks except one reduce their service capacity to μ' , we let $\mu_S = \mu$ if $|S| < K - 1$ and $\mu_S = \mu'$ if $|S| = K - 1$.

For $C \in \mathcal{C}$ and $j = 1, 2, \dots, K$, we have

$$\Gamma_p(C, C'_j) = \begin{cases} \sigma_C \left(\sum_{S: j \in S \setminus C} \frac{\mu_S \sigma_S}{|S - C| |\mathcal{N}'_S|} \right), & j \notin C \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

If the tracker does not announce all newcomers to other peers, equation (14) still holds, except for $C = \emptyset$. In this case, $|\mathcal{N}'_\emptyset| = N - 1$ but $|\mathcal{N}'_C| \leq N - 1$ for $C \neq \emptyset$ (see Section 3).

In what follows, we characterize $\Gamma_s(C, C')$ for the different strategies considered in this paper,

- *random peer, random block*: the publisher allocates capacity $U\sigma_C/N$ to serve peers with signature C , and each of the useful blocks is transferred with same probability. Then,

$$\Gamma_s(C, C'_j) = \begin{cases} \frac{U\sigma_C}{N(K - |C|)}, & j \notin C \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

- *random peer, rarest block*: let \mathcal{R}_C be the set of less replicated blocks among those which are

useful for a peer with signature C . Then, replacing $K - |C|$ by $|\mathcal{R}_C|$ in (15) we obtain

$$\Gamma_s(C, C'_j) = \begin{cases} \frac{U\sigma_C}{N|\mathcal{R}_C|}, & j \notin C, j \in \mathcal{R}_C \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

- *most deprived peer, rarest block*: let \mathcal{M} be the set of signatures of most deprived peers. Then, replacing N by $\sum_{C:C \in \mathcal{M}} \sigma_C$ in (16) we obtain

$$\Gamma_s(C, C'_j) = \begin{cases} \frac{U\sigma_C}{(\sum_{C:C \in \mathcal{M}} \sigma_C)|\mathcal{R}_C|}, & C \in \mathcal{M}, j \notin C, j \in \mathcal{R}_C \\ 0, & \text{otherwise} \end{cases} \quad (17)$$

Appendix B: Derivation of Equation (5)

Next, we derive an expression for the system throughput Ψ ,

$$\Psi = \sum_{i=0}^{J-1} \mu E[n_{G+i}] + \frac{((K-1) - (J+1))\mu + \mu'}{(K-1) - J} E[n_{G+J+}] \quad (18)$$

$$= \sum_{i=0}^{J-1} \frac{\mu}{\mu'} \sum_{l=0}^i \gamma_r(F_l) + \frac{(K - (J+2))\mu + \mu'}{\mu'} \sum_{l=0}^J \gamma_r(F_l) \quad (19)$$

$$= \frac{\mu}{\mu'} \sum_{l=0}^J \left((K-2-l) + \frac{\mu'}{\mu} \right) \gamma_r(F_l) \quad (20)$$

Equation (19) is obtained from (18) recalling that $E[n_{G+i}] = \sum_{l=0}^i \gamma_r(F_l)/\mu'$, as queue F_{G+i} is an $M/M/\infty$ queue (see Figure 5). Equation (20) is obtained from (19) after simple algebraic manipulation.

Appendix C: Details about the Lumped Model

Next, we present the algorithm used to generate the lumped state space. Algorithm 1 takes as input a state of the unlumped state space, and generates as output the corresponding state in the lumped version of the model. To this aim, it sorts the block according to their number of replicas (line 1), and reorders the block identifiers based on this ordering (line 4). The new signatures are generated (line 12) and the new state is computed (line 15).

Note that the number of states in the unlumped version of the model is $\binom{2^K - 2 + N}{N}$, *i.e.*, the number of ways of dividing N indistinguishable users into $2^K - 1$ groups, where each group corre-

sponds to a signature. The number of states in the lumped model is up to an order of magnitude smaller than the number of states in the unlumped model. For instance, for $K = 3$ and a population of $N = 20$ users, lumping decreases the number of states from 230,230 to 46,163 (see Table 2).

Algorithm 1 LUMPSTATE

Input: state σ in unlumped state space Ω

Output: state σ' in lumped state space Ω'

```

1:  $s \leftarrow$  list of block identifiers, sorted by number of replicas
2:  $i \leftarrow 1$ 
3: while  $i \leq K$  do
4:    $n(s(i)) \leftarrow i, i \leftarrow i + 1$ 
5: end while
6:  $\mathcal{C} \leftarrow$  set of subsets of  $\{1, \dots, K\}$ 
7:  $\mathcal{C} \leftarrow \mathcal{C} \setminus \{\mathcal{F}\}$ 
8: while  $\mathcal{C} \neq \emptyset$  do
9:   remove signature  $C$  from  $\mathcal{C}, C' \leftarrow \emptyset$ 
10:   $j \leftarrow 1$ 
11:  while  $j \leq K$  do
12:    if  $(j \in C) C' \leftarrow C' \cup \{n(j)\}$ 
13:     $j \leftarrow j + 1$ 
14:  end while
15:   $\sigma'_{C'} \leftarrow \sigma_C$ 
16: end while
17: output  $\sigma'$  is the lumped state

```

number of peers (N)	number of states (lumped model)	number of states (unlumped model)
5	127	462
6	243	924
7	429	1716
8	728	3003
9	1174	5005
10	1836	8008
11	2772	12376
12	4086	18564
13	5868	27132
14	8268	38760
15	11418	54264
16	15525	74613
17	20775	100947
18	27445	134596
19	35787	177100
20	46163	230230

Table 2: File with $K = 3$ blocks. Illustrating the gains due to lumping.

Appendix D: Transient Analysis of Most-Deprived Policy

Next, we consider additional results on the transient system analysis. In Figure 11 we illustrate the gains obtained when the publisher adopts the most-deprived peer policy. The larger the difference between U and μ , the more significant are the gains of using the most-deprived peer policy as opposed to the random peer selection. This is expressed both in term of the time to enter in a state where 90% of the population is in the one-club (Figure 11(a)), starting from an empty system, as well as the time to leave the state where all peers are in the one-club, and reach a state where less than half of the population has all blocks except one (Figure 11(b)).

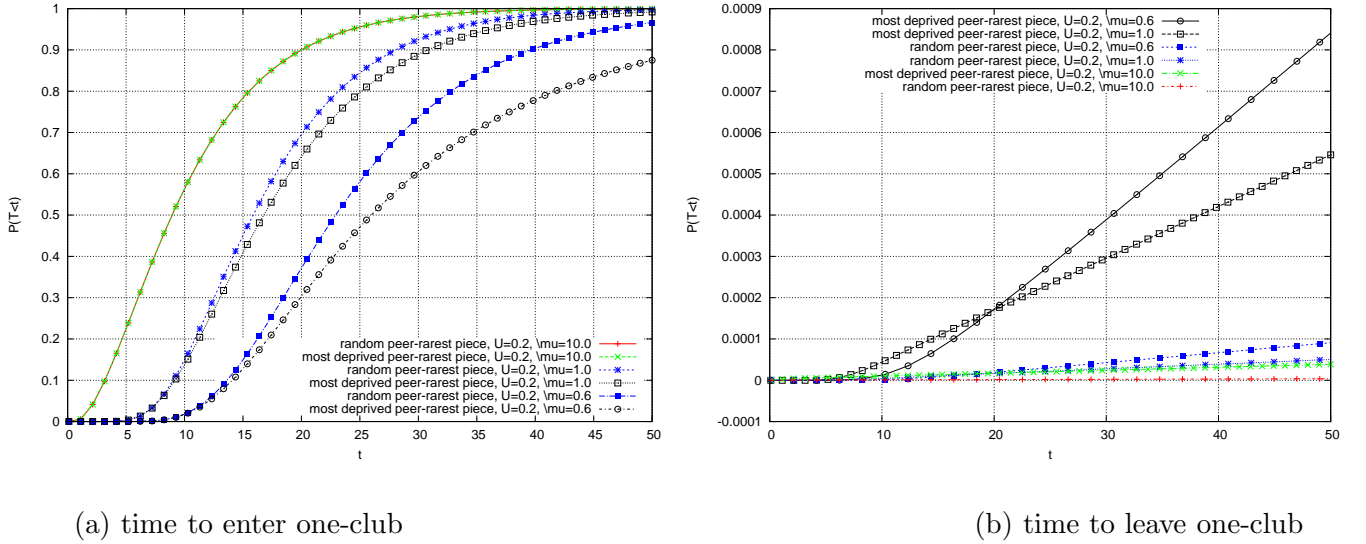


Figure 11: Transient analysis with most-deprived peer selection

Appendix E: Peers Lingering as Seeds

Next, we consider the case where peers remain in the system as seeds after completing their downloads. Let $K = 2$, $U = 1$ and $\mu = 1$. Figure 12(a) shows that when $\gamma \leq 1.2$ the throughput increases linearly as the population increases. In this case, the system is stable, in accordance to [8]. The situation is not as simple when $\gamma \geq 1.5$. When $\gamma = \infty$, Figure 12(b) shows that the throughput reaches an asymptote when the population grows, for the reasons explained in this paper. It also indicates that if the publisher replaces the random piece policy by the rarest piece policy, the gains are negligible, which again is in accordance to [8]. For more details about this scenario, refer to [17].

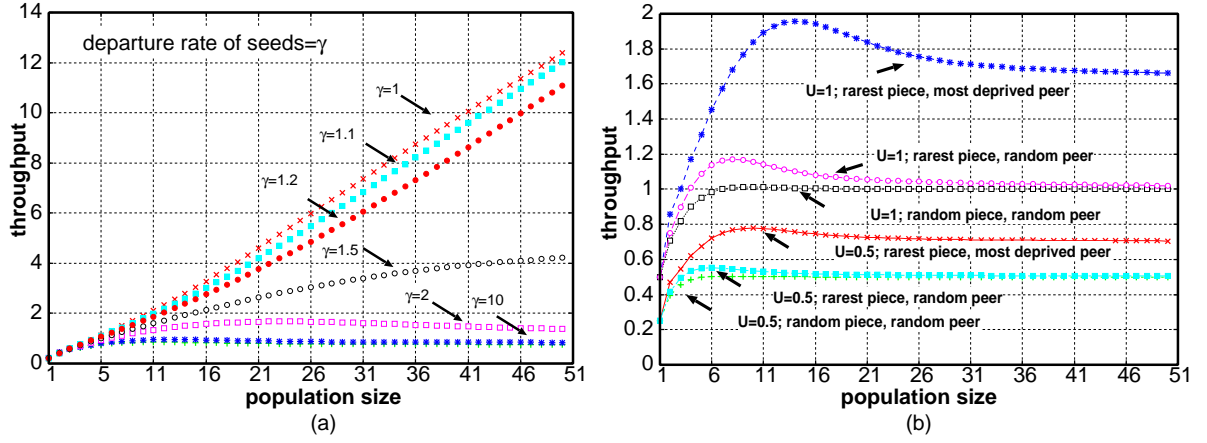


Figure 12: Scalability when peers linger in the system after completing download: (a) *peers* remain in the system as seeds for an average of $1/\gamma$ after completing *download*; (b) *peers* depart immediately.